

어느 선배  
개발자의

행복한 프로그래밍을 위한 조언

# 베츠라는 프로그래머

이제는  
개념을  
행복한 프로그래밍을 위한 '조언'

# 벤츠라는 프로그래머

초판 1쇄 발행 2013년 8월 28일

지은이 정금호

펴낸이 장성두

펴낸곳 제이펍

출판신고 2009년 11월 10일 제406-2009-000087호

주소 경기도 파주시 문발동 파주출판도시 530-1 뮤즈빌딩 403호

전화 070-8201-9010 / 팩스 02-6280-0405

홈페이지 [www.jpub.kr](http://www.jpub.kr) / 이메일 [jeipub@gmail.com](mailto:jeipub@gmail.com)

편집부 이민숙, 이 슬 / 표지디자인 미디어픽스

용지 에스에이치페이퍼 / 인쇄 한승인쇄 / 제본 광우제본

ISBN 978-89-94506-73-9 (93000)

값 13,800원

※ 이 책은 저작권법에 따라 보호를 받는 저작물이므로 무단 전재와 무단 복제를 금지하며,

이 책 내용의 전부 또는 일부를 이용하려면 반드시 저작권자와 제이펍의 서면 동의를 받아야 합니다.

※ 잘못된 책은 구입하신 서점에서 바꾸어 드립니다.

제이펍은 독자 여러분의 책에 관한 아이디어와 원고 투고를 기다리고 있습니다.

책으로 펴내고자 하는 아이디어나 원고가 있으신 분은 책에 대한 간단한 개요와 차례,

구성과 저(역)자 약력 등을 메일로 보내주세요.

[jeipub@gmail.com](mailto:jeipub@gmail.com)

어느 선배  
개발자의

행복한 프로그래밍을 위한 조언

# 베츠라는 프로그래머



정금호 지음

Jpub  
제이퍼블



이 책을 부모님께 바칩니다.

부모님의 헌신적인 사랑이 없었다면 그 무엇도 존재하지 못했을 것입니다.

감사하고 또 감사합니다.

|        |    |
|--------|----|
| 머리말    | 9  |
| 지은이 소개 | 13 |

## CHAPTER 1 프로그래밍 공부 전에 알아야 할 기초 상식 15



- 컴퓨터 \_ 17
- 운영체제 \_ 29
- 프로그래밍 언어 \_ 37
- 개발 도구 \_ 45
- 개발 지원 소프트웨어 \_ 53
- 프로그래머 \_ 60

## CHAPTER 2 나의 프로그래밍 공부법 69



- 확실한 동기를 부여하라 \_ 71
- 유사 프로그램을 벤치마킹하라 \_ 87
- 나만의 설계도를 그려라 \_ 94
- 구현에 필요한 레퍼런스를 확보하라 \_ 99
- 데드라인을 정하여 계획적으로 구현하라 \_ 108
- 완성된 프로그램은 공개하라 \_ 114
- 사용자의 피드백을 수렴하라 \_ 122

CHAPTER 3

# 나의 실패 사례 소개



127

- 갖춘 실력보다 지나친 욕심을 부리다 \_ 129
- 게임성을 고려하지 않은 기획 \_ 133
- 알면서도 떠안은 리스크 \_ 139
- 일을 끝내기 위한 추진제의 부족 \_ 144
- 기술에 대한 이해도의 부족 \_ 147
- 오픈 소스 프로젝트에 대한 경험 부족 \_ 151
- 완성하고도 출시 못 한 게임 \_ 155

CHAPTER 4

# 프로그래밍 공부를 잘하기 위한 기본 자세

159

- 끊임없는 자기계발 욕구 \_ 161
- 커뮤니케이션 능력 향상 \_ 165
- 다른 파트와의 협력 \_ 169
- 인문학적인 지식 습득 \_ 174
- 새로운 환경에 대한 빠른 적응 \_ 179
- 스트레스 해소를 위한 취미 활동 \_ 184



## CHAPTER 5

## 최근 IT 업계 트렌드

191

- 스마트폰 앱 비즈니스의 한계와 가능성 \_ 193
- 장기적인 개발 인력 부족의 심화 \_ 197
- 마이크로소프트와 어도비의 실패 \_ 202
- 반쪽의 성공, 스마트 TV \_ 206
- 클라우드와 빅 데이터에 대한 단상 \_ 210
- 윈도우즈 8, 과연 성공할 것인가? \_ 215



## CHAPTER 6

## 프로그래머로서의 꿈

219



- 소프트웨어 개발 회사 창업 \_ 222
- 30대의 꿈, 그리고 40대의 꿈 \_ 236
- 역사 시뮬레이션 게임 개발 \_ 243
- 비상업적인 오픈 소스 프로젝트 운영 \_ 246
- 개발자들을 위한 문화 조성 \_ 250
- 효과적인 소프트웨어 프로젝트를 위한 전문 컨설턴트 되기 \_ 254

집필 후기 262

찾아보기 265



필자가 처음 ‘컴퓨터’라는 것을 접한 것은 1983년경에 친구 집에서 마주친 MSX 호환 기종이었다. 그것을 계기로 1984년에 어머니 손을 잡고 컴퓨터 학원을 등록하게 되면서 애플 II+ 호환 기종으로 컴퓨터 프로그래밍 공부를 시작하였다. 1986년에는 아버지께서 당시에 엄청난 거금을 들여 애플 II+ 호환 기종 풀 세트를 사주시면서 본격적인 프로그래밍의 길을 걷기 시작했다.

1990년에는 IBM-PC/AT 호환 기종과 도트 프린터를 장만하면서 MS-DOS 프로그래밍을 시작했다. 1994년경 어머니께서 사주신 486 호환 기종과 현대 소프트웨어 공모전 상품으로 받은 486 호환 기종의 PC로 열심히 프로그래밍 공부를 했다. 대학 4학년인 1996년에는 외주 개발을 맡아 받은 계약금 300만 원으로 부모님의 허락 없이 270만 원짜리 노트북을 샀다. 1998년부터 2003년까지는 직접 게임 회사를 운영하면서 참으로 많은 PC를 조립하여 사용하였었다.

2004년 이후부터는 장비에 연연하지 않고 사용할 수 있는 컴퓨터라면 무엇이든 적응하여 써오고 있다. 컴퓨터의 사양에 대한 고집을 버리면서 필자의 프로그래밍 철학도 많이 바뀌었다. 그전에는 손에 익은 개발 도구와 개발 언어에 집착하는 경향이 있었지만, 2004년 이후부터는 개발 도구와 개발 언어에 상관없이 빠르게 적응하여 구현하고자 하는 애플리케이션을 개발하고 있기 때문이다.

근 30년 가까이 컴퓨터와 함께 길을 걷고 있는 필자는 ‘프로그래머’의 길을 선택한 것에 대해서 꽤 만족하고 있다. 내가 프로그래머라는 직업을 가지고 있기 때문에 일반적인 직업을 가진 대다수의 다른 사람에 비해 얻은 기회와 이익이 더 많았기 때문이다. 물론, 나의 아이들이 같은 길을 걷겠다고 한다면 쌍수를 들어 환영하기보다는 한 번쯤은 만류할지도 모르겠다.

요즘엔 ‘프로그래머’라는 직업이 기술직이기는 하지만, 흔히들 3D 직종이라고 인식하고 있는 듯하다. 그래서인지 일부 분야를 제외하고는 예전과 같이 좋은 인재들이 많이 유입되지 않는 것이 현실이다. 프로그래머 직종에 종사하는 인력 중에서 상대적으로 열악한 근무 환경 때문에 다른 업종으로 전직하는 경우도 많다. 게다가 국내의 개발환경에서는 30대 후반이 되면 계속 개발을 하지 못하고 관리직이 되어야 하는 고충을 토로하는 개발자들도 많다.

필자는 그렇게 힘들고 고통스럽게 ‘프로그래머’의 길을 가고 있는 분들에게는 죄송한 말이지만, 아쉽게도 그분들은 ‘프로그래머’라는 축복받은 직업에 대해서 제대로 이해하지 못하고 있는 것이 아닌가라고 생각한다. 프로그래머 또는 개발자라는 직업을 단순히 생계유지를 위해 정신 노동을 하는 직업이라고만 생각한다면 일 자체를 즐기지 못하게 되고, 그렇기에 행복 또한 느끼지 못하는 것이라고 생각한다. 같은 일이라도 어떻게 생각하고 대하느냐에 따라 천차만별의 결과를 이끌어내는 것이 세상 사는 이치이기 때문이다. 자의든 타의든 여러분이 ‘프로그래머’의 길을 걷기로 마음먹고 그 길을 걷고 있다면, 제대로 가기 위한 노력도 반드시 해야 한다.

필자가 이루고자 하는 앞으로의 여러 가지 ‘꿈’ 중에서 하나를 꼽으라고 한다면, ‘모든 프로그래머에게 꿈과 희망을 가질 수 있도록 도와주는 역할’이다. 왜냐하면, 필자는 열악하다고 하는 국내 IT 업계 종사자로서 40대에 접어드는 프로그래머이기는 하지만, 여전히 ‘행복한 프로그래머’이고 앞으로도 ‘즐거운 프로그래밍’을

할 생각이기 때문이다. 필자에게 프로그래밍이란, 책을 집필하거나 그림을 그리는 일 못지않게 충분히 ‘창의적’인 활동이다. 이러한 창작이 주는 즐거움과 고통을 기꺼이 받아들이고, 그 결과물을 다른 프로그래머들과 함께 공유하고 싶다.

이 책은 필자가 어떻게 프로그래밍 공부를 했는지, 어떤 마음으로 프로그래머의 길을 가면서 하나씩 꿈을 이루어나가고 있는지를 보여주면서 여러분이 가진 가능성과 수많은 기회에 대해서 함께 생각해보고 싶어서 집필한 책이다. 모두가 힘들다고 하는 시대를 살고 있지만, 자신에게 맞는 커다란 그림을 그리면서 열심히 살아간다면 ‘프로그래머’도 다양한 형태로 성공할 수 있다는 것을 보여주고 싶다. 다만, 철저하게 개인적인 경험을 바탕으로 집필되었기 때문에 지나치게 주관적인 부분도 많을 것이고, 자기 자랑 같은 부분도 적지 않을 것이다. 하지만 그러한 걸 모습에 집착하지 말고 우리가 어떻게 하면 ‘즐거운 프로그래밍’을 할 수 있고 ‘행복한 프로그래머’가 될 수 있을지에 대해서 함께 고민해보는 기회로 만들어주었으면 좋겠다.

필자가 초/중/고등학생이었던 시절에는 컴퓨터 잡지를 통해서 꿈을 키웠고, 대학생일 때에는 유명한 해커들이나 개발자들에 관한 책을 읽으면서 인생의 목표를 잡았었다. 부족한 책이지만, 소프트웨어 개발자를 꿈꾸는 학생들에게 조금이나마 구체적인 방향을 설정할 수 있도록 도움이 되었으면 한다. 특히, 프로페셔널 개발자로 일을 시작한 사회 초년생들에게 권하고 싶다. 여러분이 소프트웨어 개발자로 걸어가야 하는 길은 지금 여러분이 생각하는 것보다 훨씬 더 다양하고 멋진 길들도 많이 있다는 것을 느끼게 해주고 싶다. 지금 여러분이 겪고 있는 업무 환경이 전부는 아니라는 것과 우리가 행복하게 프로그래밍을 할 수 있는 환경을 만드는 것은 다른 누가 만들어주는 것이 아니라 우리 스스로 만들어야 한다는 것을 말이다.

이 책에는 개발자가 되기 위해 알아 두어야 하는 기초 상식부터 프로그래밍 공부법, 그리고 프로그래밍 공부를 할 때 필요한 자세에 대해서 필자의 경험을 바탕으로 안내하고 있다. 현재 여러분이 프로그래밍을 공부하고 있다면 필자가 제시하는 방법과 어떤 차이가 있는지를 고민해보고, 도움이 될 것 같은 요소들은 본인의 프로그래밍 공부에 하나씩 적용해보기를 바란다. 그리고 필자가 실제 소프트웨어 프로젝트를 수행하면서 겪었던 실패 사례나 최근 업계 동향에 관한 내용도 될 수 있는 대로 많이 싣고자 했다. '성공'은 수많은 시행착오를 통해서 쌓인 노하우와 경험을 바탕으로 만들어진다고 생각한다. 필자 또한 그 누구보다 많은 시행착오를 겪어왔고, 그것을 통해서 많은 것을 배우고 느낄 수 있었다. 여러분도 필자의 소중한 실패 경험을 거울삼아서 최소한 똑같은 실수는 하지 않았으면 좋겠다. 책 뒷부분에는 필자의 개인적인 꿈에 대해서 나열하였는데, 이를 통해 지금부터 여러분도 자신만의 꿈에 대해서 구체적이고 상세하게 정리하여 그것을 어떻게 이루어나갈 것인지 고민해보는 계기가 되기를 바란다.

이미 한 번 끝까지 써내려 갔었던 원고를 모두 내던져버리고, 새로운 각오와 마음으로 집필 방향을 설정하느라 적지 않은 시간이 소요되었었다. 늘 바쁘다는 핑계로 집필 속도가 터무니없이 느렸음에도, 항상 한 발 뒤에서 기다려주시고 아낌없이 많은 조언과 도움을 주신 제이펍 장성두 실장님께 감사드린다. 장성두 실장님이 계셨기 때문에 이 책이 완성될 수 있었기에 이 자리를 빌려 다시 한 번 더 진심으로 감사드린다. 더불어 어느덧 마흔이라는 나이에 들어섰지만, 뒤가 전혀 걱정이 되지 않을 만큼 항상 든든하게 힘이 되어주는 나의 가족들에게도 감사의 말을 전하고 싶다.

즐거 찾는 스타벅스에서

**정금호**

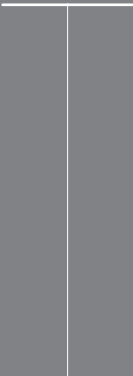
**정금호** nashorn74@gmail.com

1984년에 애플 II+로 컴퓨터 프로그래밍을 시작하였고, 대학교 1학년이었던 1993년부터 PC 통신을 이용하여 여러 가지 공개 소프트웨어와 셰어웨어를 만들어 발표하였다. 1997년부터 다수의 윈도우즈용 상용 게임과 상용 애플리케이션 및 웹 애플리케이션, 임베디드 애플리케이션 등을 개발해왔으며, 2005년부터는 윈도우즈 모바일, 아이폰, 안드로이드용 애플리케이션 및 게임을 직접 개발하거나 모바일 앱 서비스 개발 총괄 등을 하고 있다.

1995년 현대전자 소프트웨어 공모전 금상, 2009년 T스토어 애플리케이션 공모전 종합 최우수상, 2010년 SHOW 앱스토어 전국민 어플대전 어플등록왕, 2010년 T스토어 안드로이드 앱 개발 페스티벌 동상을 수상하였다.

『실전 아이폰 프로그래밍』(2011년), 『실전 안드로이드 프로그래밍』(2011년), 『애플리케이션 개발자, 윈도 모바일 매력에 빠지다』(2010년)를 집필하였고, 『애플리케이션 개발자, 아이폰 매력에 빠지다』(2010년)를 감수하였다.

# 프로그래밍 공부를 잘하기 위한 기본 자세

- 
- 끊임없는 자기계발 욕구
  - 커뮤니케이션 능력 향상
  - 다른 파트와의 협력
  - 인문학적인 지식 습득
  - 새로운 환경에 대한 빠른 적응
  - 스트레스 해소를 위한 취미 활동

전문적인 프로그래머의 길을 걷기로 마음먹었다면 가장 먼저 각오해야 할 것은 프로그래머라는 직업이 평생 공부를 해야만 살아남을 수 있다는 사실이다. 그래서 ‘프로그래밍 공부’라는 것이 처음 프로그래밍에 입문할 때에만 잠깐 거치는 과정쯤으로 생각한다면 당연히 힘들 수밖에 없다. 게다가 소프트웨어 개발 업무라는 일이 단순히 개인적인 능력만으로 처음부터 끝까지 수행할 수 있는 일도 아니므로 프로그래밍 공부를 한다는 것은 생각보다 많은 분야에 대한 이해와 노력이 필요하다는 것도 금방 알 수 있게 된다.

10년이 넘는 기간 소프트웨어 개발자로 경력을 쌓은 개발자들과 비교하더라도 어떠한 태도로 노력해왔느냐에 따라서 개인의 역량에 많은 차이가 난다. 단순히 ‘소프트웨어 개발자 노임 단계’만으로 따진다고 하면 비슷한 경력자들에게 일괄적인 급여가 책정되겠지만, 아쉽게도 현실에서는 어떤 분야에서 얼마만큼의 능력을 갖췄느냐에 따라 천차만별의 대우를 받게 된다. 이때 가장 큰 차이를 만드는 것은 단지 그저 시키는 것만 수행하는 단순 ‘프로그램 코드’이냐, 아니면 소프트웨어 개발 전체를 이해하고 그 과정에 필요한 ‘다양한 경험과 기술을 보유한 개발자’이냐 하는 점이다.

16년째 IT 업계에서 일해오고 있지만, 그나마 필자가 나름 ‘행복한 프로그래머’로서 일할 수 있는 이유도 역시 일을 대하는 기본 자세가 일반 개발자들과 차이가 있기 때문이라고 생각한다. 만일 필자 역시 단순한 밥벌이로만 소프트웨어 개발에 임했거나 그저 연봉에만 관심을 두고 나 자신의 개발에 소홀히 했다면, 과연 지금 여기까지 올 수 있었을지 의문이 든다. 어떤 분야이든 성공한 사람들을 보면 자기 일을 하면서 성취뿐만 아니라 행복까지 얻은 사람들이라는 것을 쉽게 알 수 있다. 그런 면에서 우리도 프로그래밍 공부를 잘하고, 더 나아가서는 행복한 프로그래머가 되기 위해서 지금부터라도 자신을 긍정적이고 열정적인 마인드를 가진 개발자로 바꿔보도록 하자.

## ○ 끊임없는 자기계발 욕구

늘 새로운 것에 호기심을 가져라

프로그래밍 공부뿐만 아니라 지금 하는 일에 스스로 동기부여를 잘하지 못하는 사람들은 주위에서 자신에게 비전을 제시해주지 못해 아무것도 못 하고 있다는 이야기를 곧잘 한다. 그런데 곰곰이 생각해보면, 어느 누가 우리에게 애정을 갖고 그러한 비전을 제시해주면서까지 존재하지도 않는 동기를 만들어줄 수 있겠는가? 결국, 누구든지 자신이 스스로 무엇인가에 대한 필요성을 절실하게 느껴야 하고, 그것을 이루기 위해 누구보다 열심히 무엇인가를 직접 해나갈 수밖에 없다는 이야기가 된다. 그리고 이를 위해서 무엇보다도 필요한 것이 바로 끊임없는 ‘자기계발 욕구’라고 할 수 있다.

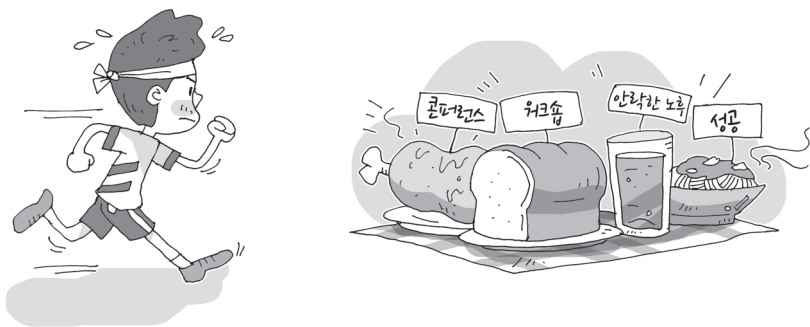
특히, 프로그래밍 공부를 할 때에는 시시각각으로 변화하는 시장의 흐름과 기술 트렌드에 빨리 적응하고, 필요하다면 이전까지 사용하던 기술은 모두 버리고 새로운 환경을 배우고 익혀야 한다. 그러므로 자기계발에 소홀하게 되면 금세 시대에 뒤떨어지는 프로그래머가 될 수밖에 없다. 실제로 현업에서 이러한 프로그래머들을 만나는 것은 그다지 어렵지 않은데, 극단적으로 이야기한다면 그들은 자기계발에 별로 관심이 있지 않을 뿐만 아니라 그 때문에 파생되는 문제에 무지해서 그러한 현상을 스스로 초래하게 되는 것이다.



또 하나 중요한 것은 무조건 자기계발에 뛰어들기보다는 자기 자신에게 꼭 필요한 방향으로 맞춤형 자기계발을 해야 한다는 점이다. 자기계발 욕구는 누구보다 넘쳐흐르지만, 어떻게 그것을 승화시켜야 하는지 잘 모를 때는 일반적인 방법의 자기계발을 시도할 수밖에 없게 된다. 가장 흔한 예가, 자기계발은 해야겠는데 어떻게 해야 할지 몰라 다른 사람들이 다니는 학원을 따라서 다니는 경우가 많다. ‘자기계발’이라는 것이 자신의 미래를 위해 투자하거나 현재 자신이 하는 일이나 처한 상황에서 좀 더 나은 결과를 이끌어낼 수 있도록 자신의 능력이나 소양 등을 개발하는 것이라고 본다면, 불필요한 교육 등으로 시간과 비용을 낭비하는 것은 오히려 역효과를 초래할 수 있다.

따라서 가장 먼저 프로그래밍 공부에 대한 목표와 비전을 확실하게 설정하는 것이 우선되어야 한다. 그것이 자신이 만들고 싶은 프로그램이든, 원하는 소프트웨어 개발 업체에 취업하는 것이든 구체적이고 이를 수 있는 목표를 가지는 것이 중요하다. 그래야 그것을 달성하기 위해서 상세한 계획을 세울 수 있고 그에 맞는 적당한 자기계발 방법을 찾을 수 있기 때문이다. 그리고 그러한 목표와 방법을 결정했다면, 그것을 제대로 이루기 위해서 현재 하고 있거나 관심을 두고 있는 일 중에서 비중이 낮은 일들은 과감히 버려야 한다. 누구에게나 주어진 시간은 같지만, 자기계발에 성공하는 사람들은 주어진 시간을 최대한 잘 활용하는 사람들이라고 생각한다. 이것도 관심이 있고 저것도 관심이 있어서 이래저래 조금씩 신경을 쓰다 보면, 결국 자기계발에 집중할 수 있는 시간이 줄어들 수밖에 없다. 그러므로 한정된 시간을 최대한 잘 활용할 수 있는 철저한 시간 관리가 필요하다.

이 이외에도 자기계발을 제대로 하기 위해 필요한 것은 나에 대한 객관적인 평가를 통해서 현재 나의 상황이 어떠한지를 정확하게 파악해야 하는 점이다. 타인에게는 냉혹하지만 자신에게는 관대한 것이 인간의 본성이기는 하지만, 나에 대해 올바르게 평가하지 못한 상태에서의 자기계발은 그 한계가 명확하기 때문이다. 그리고 혹시라도 자기계발 과정에서 발생할 수 있는 실수나 실패에 대해서 다음에 더 잘할 수 있는 계기로 만드는 태도도 중요하다. 무엇을 하든지 항상 성공할 수는 없다. 따라서 자기계발 과정에서 필연적으로 실패의 경험을 여러 번 맞닥뜨리게 된다. 이때 지나치게 좌절을 하면 중도에 포기하게 되고, 반대로 그럴 수도 있다는 식으로 가볍게 넘기는 것도 다음에 같은 실수를 하게 만들기 때문에 이러한 실수나 실패에 대한 경험을 소중하게 다루어야 한다.



꾸준하게 자기계발을 하기 위해서는 열정의 불꽃을 다시 붙일 수 있는 계기들이 있어야 한다.

---

자기계발에서 더욱 중요한 것은 ‘끊임없이’ 자기계발 욕구를 이끌어내야 한다는 것이다. 예전보다 길어진 인생과 그만큼 길어진 은퇴 시기를 살아가는 동안 한두 번의 자기계발 만으로는 분명 한계가 있기 때문이다. 그러므로 꾸준히 자기계발을 하기 위해서는 중간마다 열정의 불꽃을 다시 붙일 수 있는 계기가 있어야 한다. 필자의 경우를 예를 들자면, 다양한 콘퍼런스의 최신 기술 트렌드를 접하면서 엔지니어가 지녀야 할 열정을 다시금 지피고 있다. 예전에는 국내의 대표적인 게임 콘퍼런스에 참석해서 꺼져가던 게임 개발의 불씨를 다시 붙이기도 했고, 최근에는 윈도우즈 8 워크숍에 참석해서 또다시 윈도우즈 프로그래밍을 해보기로 마음먹게 되었다. 여러분도 자신에게 맞는 방법을 찾아 지속적인 자기계발 욕구가 생기도록 마음에 채찍질을 가할 필요가 있을 것이다.

---

## ● 커뮤니케이션 능력 향상

다른 사람의 이야기를 잘 듣고, 내 생각을 잘 전달하라

소프트웨어 개발자는 영업을 하거나 고객을 직접 대하는 직종이 아니라서 커뮤니케이션에 대해서는 소홀해도 된다고 말하는 사람들도 있다. 아쉽게도 커뮤니케이션 능력이라는 것은 고객을 상대할 때만 필요한 것이 아니라 같이 일하는 동료나 상사, 또는 후배들과 커뮤니케이션할 때에도 필요하다. 커뮤니케이션은 사람과 사람의 상호작용이므로 태어날 때부터 커뮤니케이션을 잘하는 인간이란 존재하지 않을 것이다. 선천적으로 타고난 성품이나 성격이 커뮤니케이션을 잘하는 데 도움이 될 수는 있겠지만, 기본적으로 커뮤니케이션 능력은 후천적인 노력으로 개발될 수 있다고 생각한다. 커뮤니케이션을 잘하는 것도 역시 많은 시행착오를 거치면서 경험을 쌓고 그 기술을 발전시켜야 하므로 부단한 노력과 인내가 필요할 뿐이다.

비단, 프로그래머뿐만 아니라 한 분야를 집중적으로 파고들어서 전문적인 지식과 능력을 키워야 하는 분야에서 일하는 사람들의 공통적인 특징은 커뮤니케이션 능력이 상대적으로 떨어진다는 것이다. 프로그래밍 업무를 잘하는 개발자 중에는 커뮤니케이션 능력이 부족하여 본인의 업무 능력을 제대로 인정받지 못하거나, 심지어는 불이익까지 당하는 경우도 어렵지 않게 볼 수 있다. ‘말 한마디로 천 냥 빚을 갚는다.’라는 말처럼 커뮤니케이션을 잘하게 되면 자신이 지닌 능력을 더욱 돋보이게 하고, 진행하

는 일도 좀 더 수월하게 수행할 수 있다. 따라서 커뮤니케이션 능력을 향상하기 위해서는 별도의 노력을 기울여야 할 것이다.

필자는 고등학교 시절에도 컴퓨터 동아리 활동을 했었지만, 대학교에 진학 후 컴퓨터 동아리를 직접 만들고 회장을 하면서 나 자신의 ‘커뮤니케이션 능력’에 문제가 있음을 알아차렸다. 워낙 추진력이 강하고 리더십이 있는 편이라 회장 소임을 수행하는 것 자체는 문제가 없었지만, 동아리 회원들과의 커뮤니케이션 방법에서는 미숙한 점이 많았었다. 이를 깨닫고 난 후부터 필자는 커뮤니케이션을 잘하는 방법에 대해서 진지한 고민을 시작하게 되었다.

그런 이유로 대학생생활을 하면서 리더십이나 커뮤니케이션에 대한 많은 고민을 하고 이와 관련된 책도 많이 읽게 되었다. 덕분에 20대부터 사업을 시작했음에도 어떤 일을 하든지 커뮤니케이션이 제대로 안 되어 발생하는 문제는 없었던 편이었다. 그러던 중 우연히 필자에 대한 제3자의 충격적인 평가를 듣게 되었다. ‘그 사람은 말은 정말 잘하는데, 제시한 조건이 마음에 안 들었다.’라는 이야기였는데, 여기서 필자가 충격을 받은 것은 다름이 아니라 ‘말은 정말 잘하는데’라는 문장이었다. 이것을 다른 관점에서 보면, ‘말만 그럴듯하게 잘하는 사람’으로 볼 수도 있다는 생각이 들었기 때문이다.

그래서 그 이후부터는 내 이야기를 하기보다는 먼저 상대방의 이야기를 열심히 듣기로 마음먹었다. 물론, 아무리 그렇게 마음먹었더라도 필자 역시 사람인지라 어느 사이에 내 이야기만 하는 나를 보고는 얼른 입을 다물게 되지만, 먼저 듣고자 노력하니 이전보다 더 나은 관계가 형성되고 커뮤니케이션도 원활해지는 걸 느낄 수 있었다. 아무래도 상대방의 말을 열

심히 듣다 보면 상대방이 어떤 생각을 하고 있는지, 무엇을 원하는지 쉽게 파악할 수 있게 된다. 또 하나, 단순히 열심히 듣는 것 말고도 중간마다 적절하게 상대방의 말에 피드백을 해주는 것도 중요하다. 말하는 사람 입장에서는 자신의 이야기에 상대가 지속해서 반응을 보이면 경청하고 있다고 느끼게 된다. 그래서 오히려 상대의 의견을 더 존중해주기도 한다.



경청은 상대방과의 관계를 돈독히 해줄 뿐더러 커뮤니케이션도 원활하게 해준다.

이렇게 다른 사람과의 커뮤니케이션 어느 수준에 이르게 되니 일반적인 업무 미팅이 아닌 사업 제안 발표나 강의/강연도 그다지 어렵지 않게 할 수 있게 되었다. 다수에게 발표하거나 강연하는 것 역시 일방적인 커뮤니케이션이 아니라 상호 커뮤니케이션의 원칙이 그대로 적용되기 때문이다. 제안 발표를 위해서 훌륭한 발표 자료를 준비하는 것은 반드시 필요하지만, 그렇게 준비된 발표 자료를 심사자들에게 그대로 읽어주는 것은 발표라고 할 수 없다. 발표 자료를 미리 준비한 시나리오대로 보여주면서 핵심

내용에 대한 설명을 조리 있게 해나가되, 심사자들이나 청중의 반응을 살펴 보면서 적당하게 내용을 가감하고 관심이 있어 하는 부분에 대해서는 좀 더 상세하게 설명하는 것이 좋다. 특히, 심사자들이나 청중이 궁금해하는 부분에 대한 답변은 자신 있는 태도로 자신이 아는 선에서 명확하게 전달해야 한다.

---

영업 파트 담당자들과 협의가 잘되는 개발자, 기획자나 디자이너와 커뮤니케이션이 잘 되고 팀워크를 잘 이루는 개발자, 중요한 프로젝트의 제안 발표를 잘하는 개발자 등등. 이러한 개발자가 될 수 있다면 여러분의 가치는 더욱 올라갈 수 있지 않을까? 언제까지나 말이 안 통하는 '고집불통 개발자'라는 말을 들으면서 살 것인지에 대해 진지하게 고민해보도록 하자. 여러분이 조금만 더 노력한다면 '인정받는 개발자'가 되는 것은 생각보다 어려운 일이 아니다.

---

## ○ 다른 파트와의 협력

다른 부서 사람은 적이 아니다

보통, 속하고 있는 회사나 조직의 분위기에 따라서 각 파트별로 여러 형태의 관계를 유지하게 된다. 예를 들어, 디자이너들의 파워가 센 회사에서는 개발자들의 개발 과정에 디자이너들의 의견이 크게 영향을 미치게 되고, 개발자들의 영향력이 큰 조직에서 디자이너들은 그저 개발자들이 지시하는 대로 디자인 작업을 해야 하는 때도 있다. 이처럼 프로젝트를 수행하는 데 연관된 부서들이 조화롭게 협력하는 구조가 아니라 특정 부서가 일방적으로 주도하는 형태를 띠게 되면, 프로젝트의 진행 과정에서부터 문제가 많을 수밖에 없다.

따라서 아무리 이렇게 정치적으로 복잡한 관계가 있다고 하더라도 다른 부서(파트)들과 협력해서 일할 때에는 가능한 한 좋은 결과를 이끌어내기 위해 (또는 좋은 제품을 만들어내기 위해) 적극 협력하는 태도가 가장 중요하다. 부서 간의 관계도 결국 사람 간의 관계처럼 서로가 얼마나 노력하느냐에 따라 결과가 다르게 나올 수 있기 때문이다. 그러므로 될 수 있으면 불필요한 대립 구조를 만들기보다는 서로 상생할 수 있는 분위기를 만들어서 서로에게 도움을 줄 수 있는 방향을 모색해야 한다.





프로젝트는 부서 간, 팀원 간의 협업이 절대적으로 필요하다.

## 상대의 입장을 이해하고자 노력해야 한다

일하다 보면 본의 아니게 상대방이 오해할 수 있는 일들이 쌓일 수 있게 된다. 이런 때는 먼저 상대 부서를 이해하려고 노력하는 것이 우선되어야 하고, 반대로 우리로서도 놓치는 일이 없었는지를 재확인하는 기회로 삼아야 한다. 필자는 팀원들에게 일단 다른 부서와 일하다가 문제가 생기면, 무조건 ‘내 탓이오.’를 외치면서 내가 잘못된 부분이 없는지를 되짚어 보라고 한다. 설령, 상대 부서의 담당자가 명백한 실수를 저질렀다고 하더라도 그래도 내 실수라고 생각하면서 발생한 문제를 함께 해결하고자 노력한다면 다른 부서와의 협력은 그리 어려운 문제가 아니게 된다.

## 부서장 간의 커뮤니케이션이 중요하다

문제가 있다고 판단될 때에는 부서원(팀원)들 간에 직접 분쟁이 일어나지 않도록 부서장들끼리 해당 문제에 대한 협의를 신속하게 진행해야 한다. 어떤 문제이든 부서 간에 문제가 생기는 것은 양쪽의 책임자들에게 1차적인 책임이 있다. 따라서 부서장들끼리 부담 없이 커뮤니케이션할 수 있는 채널을 만들어놓고 해당 채널을 이용하여 수시로 협의하는 것이 필요하다. 이렇게 중요한 문제를 부서원(팀원)들에게 전가하고 소홀히 한다면, 그것은 부서장의 ‘업무태만’이라고 할 수 있다.

## 업무 협의는 가급적 이메일로 한다

같이 업무를 수행하다가 나중에 문제가 생겼을 때 가장 곤란한 것은 해당 문제의 원인을 찾을 수 없을 때다. 중요하든 중요하지 않든, 구두 협의를 통해서 업무를 수행하게 되면 서로의 기억력을 놓고 티격태격할 수밖에 없다. 따라서 사소한 협의 내용도 가능하면 이메일로 관련된 사람들에게 모두 공유해놓는 습관이 필요하다. 불분명한 기억력에 의존하기보다는 명확하게 남아있는 문서로 시시비비를 가리는 것이 서로에게 상처를 주지 않을 수 있는 가장 현명한 방법이다.

## 이슈 트래커를 활용한다

소프트웨어 개발 시에 사용되는 ‘이슈 트래커’는 같은 부서 내의 다른 개발자들과 협업할 때뿐만 아니라 다른 부서와 협업할 때에도 큰 도움이 된다. 필자는 항공사진 서비스를 개발할 때 개발팀과 지도 제작팀 사이의 업무를 별도로 개발한 이슈 트래커를 이용하여 업무 협조 및 모니터링을

하게 만들었다. 예를 들어, 지도 제작팀에서 어떤 지역의 지도 작업이 끝났으니 서비스에 적용해달라는 이슈를 개발팀 쪽에 던지면, 담당 개발자가 만들어진 지도 데이터를 검수하고 문제가 없으면 서비스에 적용하는 것이다. 혹시라도 지도 데이터가 잘못 만들어졌거나 실수가 있었다면, 개발자가 다시 지도 제작팀 담당자에게 어떤 문제가 있으니 재작업을 해달라고 할 수 있다. 이러한 과정은 모두 히스토리가 남고 서로에게 요청한 사항에 대해서 진행 여부를 이슈 트래커를 이용하여 실시간으로 확인할 수 있어서 굳이 서로의 감정을 다치면서 일할 이유가 없어진다.

## 같이 어울리는 자리를 마련한다

아무래도 ‘일’이란 사람이 하는 것이므로 가능하면 다른 부서의 담당자들과 함께 어울리는 시간을 많이 가지는 것이 무엇보다 중요하다. 일반적으로, 사람과 사람이 친해지기 위해서 하는 행동들(가벼운 운동 함께하기, 함께 식사하거나 술 한잔하기 등)을 다른 부서 담당자들과 함께 하게 되면, 그들도 나와 같은 감정을 가진 사람이라는 것을 다시금 느끼게 될 뿐만 아니라 담당자마다의 성격을 좀 더 파악할 수 있게 된다. 이렇게 노력하여 알게 된 상대방의 성격이나 일하는 스타일을 고려하여 일하면, 상대방을 몰랐을 때와 비교해 더 많은 시너지 효과를 얻을 수 있게 된다. 그러므로 부서 간의 회식이나 회사 전체의 워크숍도 결국 ‘일의 연장선’이라고 할 수 있다.

---

만일 당신이 속한 팀이 ‘개발팀은 말이 통하지 않아.’ ‘개발팀이 항상 문제야.’라는 식의 말을 듣는다면, 상대방을 무조건 비난하고 방어하기에 급급하기보다는 그동안 해왔던 업무 방식에 대해서 반성을 하는 것이 좋다. 물론, 모든 사람을 이해시키고 납득하게 만드는 것이 쉽지 않기는 하지만, 무슨 이유에서든 분명히 어떤 문제가 있기 때문에 상대방으로부터 그러한 말을 듣게 되는 것이다. 함께 좋은 결과를 만들기 위해서는 항상 다른 이의 말에 귀를 기울이고, 항상 자신을 되돌아볼 줄 아는 태도가 필요하다. 나 자신부터 모든 것은 다 내 탓이라고 생각하고 노력하게 되면, 자연스럽게 상대방도 조금씩 바뀌게 된다.

---

## ● 인문학적인 지식 습득

단지 기술밖에 가진 게 없다면 잃는 것이 많을 수밖에 없다

하드웨어뿐만 아니라 소프트웨어 개발 분야 역시 공학을 전공한 엔지니어들이 주도하여 수행하는 분야이다 보니 아무래도 모든 문제에 기술적인 마인드로만 접근하는 경향이 강할 수밖에 없다. 그래서 PC가 태동하여 전성기를 구가할 때까지는 이렇게 기술적인 관점에 치중되어 발전되어 왔다. 그러다가 어느 정도 하드웨어/소프트웨어 기술이 평준화되면서부터 단순히 기술적으로 접근하는 것보다 여러 가지 다양한 분야의 감성을 접목하고 융합하는 방향으로 자연스럽게 발전하기 시작했다.

아마도 이러한 기술과 감성의 결합에 대해서 강박적으로 접근한 사람이 바로 ‘스티브 잡스’일 것이다. 애플의 혁신적인 제품을 통해 우리는 단순한 엔지니어링 관점에 인문학적인 요소를 더하면 훨씬 더 훌륭한 제품을 만들 수 있다는 것을 배울 수 있었다. 그래서 요즘 엔지니어들에게는 일부러 인문학적인 지식을 습득하는 것 자체가 또 하나의 자기계발 방법으로 자리를 잡고 있다. 이러한 인문학이 더 중요한 이유는, 단순히 더 좋은 제품을 만들기 위해서뿐만 아니라 소프트웨어 개발에 종사하고 있는 우리 개발자들 스스로가 더욱 풍족하고 나은 삶을 살기 위해서도 반드시 필요하기 때문이다.

장석주 시인은 『일상의 인문학』이라는 자신의 책 서문에서 다음과 같이 이야기하고 있다.

당신은 안녕한가? 당신의 안녕함이 누군가의 안녕하지 못함을 담보로 얻어진 것이라면 그것은 진정한 안녕함이 아니다. 그것은 처벌이나 단죄가 없더라도 실효된 악이다. 그런 까닭에 삶이 드난살이라 할지라도 맑고 순정한 눈빛을 잃어서는 안 된다. 저마다 돈 되는 것들에 정신이 팔려 정작 삶은 이리 치이고 저리 치여 이토록 진부함 속에 방치되고 있는 걸 보면 인문학이 위기라는 것은 빈말이 아닌지도 모른다. 인문학은 본질에서 삶을 살찌우고 풍요하게 만든다. 그것은 밥을 주고 실용으로 써먹는 데 소용이 닿지 않을지 모르지만, 우리 삶을 잘 누리는 데 기여하는 학문이다.

필자는 장석주 시인의 이야기에 전적으로 동감한다. 물론, 비단 IT 업계의 개발자들에게만 한정되는 이야기는 아니겠지만, 지금의 소프트웨어 개발자들에게서 ‘인문학 정신’이 모자라 메마르고 타산적인 면모가 강하게 드러나고 있기 때문이다. 어찌 된 일인지 모두가 사소한 부분조차 ‘양보’하지 못하고 한 톨이라도 손해를 보지 않겠다는 태도로 일관하고 있으며, 함께 일하고 있는 다른 사람에 대한 ‘배려’는 눈곱만큼도 찾아보기가 어렵다. 이것이 요즘 세대에는 당연한 현상이라고 한다면 이처럼 비인간적인 시대도 없을 것이다.



엔지니어링 지식에 다양한 인문학적 요소가 더해지면 혁신적인 제품이 나올 확률이 더 높다.

적지 않은 돈을 받고 일하면서도 진행하고 있는 일에 대해서는 누구보다 방관자적인 태도로 임하는 개발자, 급여를 한두 푼 더 준다고 미련 없이 다른 회사로 이직하기 위해 퇴사 1~2주 전에 일방적으로 통보하는 개발자, 누구보다 뛰어난 개발자가 되고 싶다고 말하면서 그것을 위해 감내해야 하는 과정은 강하게 거부하는 개발자, 고객이나 의사결정권자의 의견을 존중하지 않고 자신의 디자인 철학만을 떠들어대는 디자이너, 헌신적으로 개발하고 있는 개발자들의 노력 따위에는 관심이 없고 자신의 체면만을 중시하는 관리자, 이미 많이 고생을 해봤기 때문에 또다시 당하지 않겠다고 선을 긋고 일하는 개발자 등등 우리 주위에서 어렵지 않게 볼 수 있는 사람들이다.

처음에는 이들의 말과 행동이 전혀 이해가 되지 않았었는데, 지금은 그들이 자기 자신에 대한 성찰의 시간을 제대로 가지지 못해서 발생한 것이라는 걸 알게 되었다. 어떤 결과에 대해 나 자신에게서 문제의 원인을 찾기보다는 손쉽게 다른 사람과 주위 환경을 탓하고, 그러한 과정을 반복하다 보니 나 자신의 문제를 찾아서 고치기보다는 주위 환경만 바꾸는 방법으로 문제를 회피하게만 되는 것이다.

필자가 까다로운 고객사 담당자를 만나서 근 2년간 엄청난 스트레스를 받으며 프로젝트를 진행하던 시기의 일이다. 당시에 같이 일하던 그 담당자는 회의를 하면 서너 시간은 기본이었고, 매번 새로운 아이디어를 가지고 와서 쏟아내는 것이 일이었다. 프로젝트를 진행하면 할수록 산으로 간다는 느낌을 넘어서서 절대로 빠져나올 수 없는 늪에 빠진 기분이었다. 그래서 그 담당자와 미팅하기 위해 고객사로 향할 때마다 참으로 암담한 기분이 들었었다. 그 무렵 우연히 코이케 류노스케 스님의 『생각 버리기 연습』이라는 책을

읽게 되었다. 책을 읽으면서 문제의 원인은 그 담당자가 아니라 나 자신이  
었다는 간단한 진리를 깨닫게 되었고, 그때부터는 이전과는 다른 관점에서  
그 담당자와 커뮤니케이션을 할 수 있었다. 프로젝트도 수월하게 진행된 것  
은 물론이다.

필자는 어렸을 적부터 책을 무척 좋아했었기 때문에 초등학교 때는 시립  
도서관을 다니는 것이 취미였었고, 중학교 때는 틈만 나면 헌책방을 들락  
거렸다. 지금도 3개월마다 10~15권 정도 다양한 분야의 책을 구매해서 틈  
날 때마다 읽고 있다. 이렇게 많은 책을 읽다 보면 다양한 사람들의 생각  
과 경험을 배울 수 있고, 그중 일부는 실제 생활이나 업무에 적용해볼 수  
도 있게 된다. 그러다 보니 나 자신도 마음의 양식이라는 것이 조금은 생  
긴 것 같고, 치기 어린 20대에 비하면 상대적으로 많이 성숙해졌다고 느  
껴진다. 덕분에 다른 사람에 비해서 더 많은 마음의 여유를 가질 수 있게  
되었고, 그만큼의 인내심과 끈기도 가지게 되어 사회생활을 하는 데에도  
큰 도움이 되고 있다.

실력이 형편없으면서 커뮤니케이션도 어려운 개발자, 그럭저럭 실력은 있  
지만 성격이 까다로운 개발자, 좋은 개발자임에도 다른 사람들로부터 왕  
따를 받는 사람 등 저마다 다른 성격과 실력의 개발자들과 함께 지극껏  
수많은 프로젝트를 비교적 무난하게 수행할 수 있었던 것도 이러한 노력  
덕분이었다고 생각한다. 인문학적인 지식을 습득하는 것은 팀원들을 이  
끌 때만 도움이 된 게 아니다. 사업 파트너나 직장 상사들과 함께 일할 때  
에도 큰 힘을 발휘해서 오히려 나 자신의 가치를 올리거나 업무 능력을 인  
정받는 데 도움이 되기도 했다. 자신이 받는 월급에 맞게 자신이 맡은 부  
분만 잘하면 일을 잘하는 것이고, 그것이 자신이 더 나은 대우를 받을 수



있는 근거라고 생각하는 사람들 관점에서는 전혀 이해할 수 없는 혜택을 누릴 수 있게 된 것이다.

---

인문학을 익히기 위해서 책을 많이 읽는 것도 중요하지만, 본인의 양식을 풍요롭게 만들 수 있는 좋은 책들을 잘 찾아서 읽는 노력이 더 중요하다. 그런 점에서 필자는 개인적으로 존경하는 분들이 추천해주는 책을 좋아한다. 책이란 게 자기 자신이 정말로 감동하지 못했다면 선뜻 추천해줄기가 어렵기 때문이다. 지금까지 여러분이 읽은 책 중에서 추천해줄 만한 책은 무엇이 있을까 생각해보자. 그리고 주위 사람들에게 좋은 책을 추천받아서 읽어보도록 하자. 책은 마음의 양식이며, 여러분이 조금 더 좋은 개발자가 되는 데 큰 도움이 되는 밑거름이다.

---